International Journal of Material Forming (2018) 11:867–878
https://doi.org/10.1007/s12289-017-1396-x

ORIGINAL RESEARCH

Methodology for metamodelling of microstructure evolution: precipitation kinetic case study

Piotr Macioł[1] · Danuta Szeliga[1] · Łukasz Sztangret[1]

Received: 18 September 2017 / Accepted: 27 December 2017 / Published online: 15 January 2018

Abstract
Numerical modeling is an important tool assisting in the designing and optimization of the production technology. The highest predictive capabilities are offered by multiscale modeling. The most important limitation of its wide application is computational cost. One of possible solutions is application of metamodels for fine scale modeling. In this paper, a systematic approach to development of metamodels is presented. All necessary steps, analyzing the model, selecting the metamodel inputs and outputs, gathering the training and testing datasets, choosing a metamodelling technique, training and testing the metamodel are described with a scientific background and practical examples. Development of the exemplary metamodel, replacing thermodynamic modeling of precipitation kinetic is presented.

Keywords Multiscale modeling · Metamodels · Sensitivity analysis · Kriging · Surogate model · Precipitation kinetic

Introduction

Most modern metallic products are highly complex and sophisticated. In order to achieve the desired material properties, manufacturers aim to control the development of their products down to the microscopic scale. The production processes usually consist in a succession of thermo-mechanical steps during which the microstructure of the material evolves in a complex manner. Numerical modeling is used to assist in the designing and optimization of the production technology, but modeling the thermo-mechanical treatments (TMT) of metallic alloys requires the consideration of various phenomena happening in several spatial and temporal scales.

Complex multiscale models for metallic materials usually combine a macroscopic solution of heat transfer and deformation with finer scale models for the underlying microscopic phenomena such as dislocation density evolution, recrystallization and phase transformations. The Agile Multiscale Modeling Methodology (AM3) [1–4] was developed in order to ease the development of such models. It defines a standard model structure in which a macroscopic component interacts with several microscopic modules according to a set of user-defined rules. Although the AM3 simplifies the model structure, the microscopic modules can remain quite complex and consuming in terms of computer power. Modeling the kinetics of phase transformations, for example, often requires the use of a third party, resource demanding thermodynamics solver, that cancels the gains of the AM3.

In order to deal with such demanding microscopic models, the concept of metamodelling is introduced in the AM3. Metamodels are designed to cut down the complexity of the models to which they refer while keeping an acceptable degree of accuracy and reliability. The models are usually built to represent physical mechanisms, while the metamodels are representations of the models, numerically built to reduce their complexity. In this paper, a metamodel is built on top of a microscopic precipitation model for second-phase particles, within the framework of an AM3-structured multiscale model for TMT of metallic materials [4].

✉ Piotr Macioł
pmaciol@agh.edu.pl

[1] Faculty of Metals Engineering and Industrial Computer Science, AGH University of Science and Technology, al. A. Mickiewicza 30, 30-059 Kraków, Poland

Agile multiscale modeling technology (AM3)

The AM3 is a design methodology for structuring complex multiscale models. It aims to reduce the programming effort required to develop fully coupled models, and limit the computing resources needed by the model. The multiscale model consists in a macroscopic module linked to several lower-scale modules. The state of the model is continuously tracked by a Knowledge Base System and relayed to a control unit that decides on the run whether to use a lower scale model, and which one to use, in order to provide the best compromise between reliability and computing time. The decisions made by the control unit are based on a set of user defined rules that provide, among others, the range of applicability of each fine-scale model.

Fine scale models classically feature local variables, averaged over a Representative Volume Element (RVE) or a Statistical Volume Element (SVE) [5, 6]. In a coarse domain, an instance of the fine scale model needs to be run for each computational point. If the fine scale model is computationally demanding, the overall simulation will require large computational resources and/or a long computational time, even in a parallel or a cloud environment. Instead, the AM3 framework introduces adaptability, meaning that the structure of the multiscale model can adapt to the actual conditions – phenomena expected to be present in a process, required reliability and available computing power. Fine scale models are used only when and where they are needed. In the original AM3 approach, fine scale models could just be turned on or off. In presented research, several models may be available for the same variables, differing mainly by the computational costs and the reliability but also by the range of applicability. The control unit can choose the most relevant one. For example, there could be two models of recrystallization, one based on Internal Variable (IV) approach [7] and a second based on Phase Field method [8]. The first is much less computationally demanding, however reliability of the IV-based model worsens rapidly when process conditions become different from the ones for which the IV model had been calibrated. Hence, in some range of process parameters both models provides the same reliability. In other range, the IV model is clearly non-applicable. In both ranges an optimal choice of the model is clear – IV-based in the first one and Phase Field in the second. However, there is still a range when both models may be used and their relevancy depends on process parameters but also required reliability and available computing power. In the present state of the AM3 framework, the mentioned above KBS system is able to decide is fine scale model necessary in particular computational point and if yes, which fine scale model is most relevant in particular conditions and requirements.

The metamodelling idea

A computer model (hereinafter referred to just as "model") is the algorithms and equations used to capture the behavior of the process.

Simulating real processes usually requires complex models, which in turn require significant computing resources. Therefore, it makes sense to look for approximate methods that allow a reduction of computing time while keeping a required accuracy. The main idea of metamodelling lies in the assumption that a metamodel (called also the surrogate model) approximates a model previously defined [9]. The metamodel must accurately correspond to the model, but its output has to be returned with a significantly lower computational effort than using the model. Metamodelling is a construction of an approximate model of the analyzed process on the basis of the model of the same process. In other words, the metamodel is *the model of the model*. The accuracy of the metamodel, usually verified using statistical methods, depends on the applied metamodelling technique and the number of measured data points used to parametrize the metamodel. Generally, the larger the dataset used to train a metamodel, the greater the degree of accuracy achieved. However, producing a data point requires a time consuming simulation using the model. Therefore it is advisable to restrict the number of data points. This can be achieved using Sensitivity Analysis (SA) [10, 11] or Design of Experiments (DoE) [12]. Metamodels can be built using several approaches. Two of them will be considered within the scope of this work: the Artificial Neural Network method and the Kriging method [13].

The majority of metamodel applications concerns single-scale models, but Hambli and Barkaoui presented the application of a metamodel in multiscale modeling [14, 15]. Combining a knowledge-based adaptive multiscale model with metamodelling is a promising idea in the context of the AM3 approach. If metamodels of the fine scale models were to be added to the set of available choices, they would be promoted by the KBS due to their lower computational requirements, if they only provide acceptable reliability. However, this approach is more challenging than classical metamodelling. AM3 is a generic methodology, independent of any particular numerical problem. Hence, the issue is not to develop a metamodel for particular phenomena, but rather to develop *a methodology, supporting the development of metamodels*. Also, because

AM3 promotes adaptability, any metamodel developed in that context must be described with corresponding rules and facts in the KBS concerning their conditions of applicability and reliability.

The general workflow

The general workflow of development of a metamodel process consists of six steps:

1. Analyzing the model
2. Selecting the metamodel inputs and outputs
3. Gathering the training and testing datasets
4. Choosing a metamodelling technique
5. Training the metamodel
6. Testing the metamodel.

The preliminary analysis of a model provides some information about its character. Knowledge about the model linearity and a time invariance is very useful during the development of a metamodel. Particularly, if the model takes into consideration the history of the process, a different metamodelling technique must be used. Input selection for the metamodel can be done arbitrarily by the designer [16] or based on a sensitivity analysis [17] (described in the next section). The next step consists in gathering training and testing datasets, for which multiple simulations are required. The simulations done during the SA can additionally be used for that purpose. Basing on the character of the model (linearity, time invariance etc.) and the size of the training dataset, a technique is chosen to build the metamodel. There are no clear guidelines on how to make this decision. Some hints can be found in [13]. A metamodel training method is dedicated to each specific metamodel type. Final testing of the developed metamodel is performed with statistical evaluation methods.

Sensitivity analysis

Sensitivity analysis (SA) helps investigating the influence of the input parameters on the model outputs [10, 11]. A physical phenomenon usually is described as a system of mathematical equations (model of a problem) and the parameters of the model (the input and coefficients) characterize the phenomenon. The main objectives of a sensitivity analysis are threefold: i) determine the regularity and reliability of the model to verify its structure, ii) determine the parameters sets having the lowest and highest impact on the model and iii) determine possible correlations between model parameters. In this paper, two algorithms of SA are used: the method called Factorial Design (FD) [10] and an approach based on the Morris Design (MD) [18]. The main assumaptions and the algorithms steps are presented briefly below.

In the two-level FD, for each parameter the upper limit (marked as "+") and the lower limit (marked as "-") is specified and they define two-levels of the parameter. The points in the algorithm are generated starting with all low levels and ending levels. It means that for k parameters, model is run 2^k times. The average response from high level runs is contrasted with the average response from the low runs to determine the effect:

$$Effect_{FD} = \frac{\sum y^+}{n^+} - \frac{\sum y^-}{n^-} \tag{1}$$

where y is the model output, "+", "-" is the upper/lower limit of the parameter range, respectively, n is the number of model simulations at each level.

In the MD algorithm, the term *main effect* of parameter is introduced and it is determined by computing a number of local measures at different points in the input space and next estimated by mean value and standard deviation. Let $\mathbf{x}$ be an n-dimensional vector of model parameters x_i. The primary assumption of the algorithm is that all x_i components are defined on $[0, 1]$ interval. In most practical problems x_i components are of various physical units and the parameters have to be rescaled to $[0, 1]$. The conversion is necessary to compare the results obtained for various parameters. It is feasible only if estimated elementary effects are expressed with the same units for all parameters. Let the components $x_i, i = 1, \ldots n$, accept k values in the set $\{0, 1/(k-1)/k-2), \ldots, 1\}$. Then the parameters domain $\Omega \subseteq \Re^n$ forms an n-dimensional k-level grid. Let Δ depend on k and describe the side length of the grid element:

$$\Delta = \frac{1}{k-1} \tag{2}$$

The *elementary effect* ξ_i of the i^{th} parameter at a given point $\mathbf{x}$ calculated for y model output is defined as:

$$\xi_i(\mathbf{x}) = \left| \frac{y(x_1, \ldots, x_{i-1}, x_i + \Delta, x_{i+1}, \ldots, x_k) - y(\mathbf{x})}{\Delta} \right| \tag{3}$$

where $\mathbf{x}$ is any value in the Ω domain such that the perturbed point $\mathbf{x} + \Delta$ is also in Ω. A finite distribution F_i for each parameter x_i is obtained by sampling $\mathbf{x}$ in Ω. The number of elements of F_i is equal to $(k-1)k^{n-1}$.

Distribution F_i of elementary effects is described by *mean* μ and *standard deviation* σ. A mean characterizes the sensitivity of the model output with respect to the i^{th} parameter. A high mean indicates that the parameter is

important and it substantially influences the output. A high standard deviation implies that the parameter interacts with other parameters or its effect to the model is nonlinear.

Comparing MD and the two-level FD methods of estimation the parameter effect to model output, it is observed, that FD is computationally cheaper than MD: for three parameters, FD requires $2^3 = 8$ runs of the model while MD needs 16 runs to obtained equivalent results.

In most metamodelling methods, keeping inputs with a low sensitivity index might decrease the reliability of the metamodel. Increasing the number of input parameters also exponentially increases the training time and the minimal cardinal of the training set. In the context of metamodelling, a sensitivity analysis is used to optimize the structure of the metamodel by comprehensively reducing the number of inputs [19].

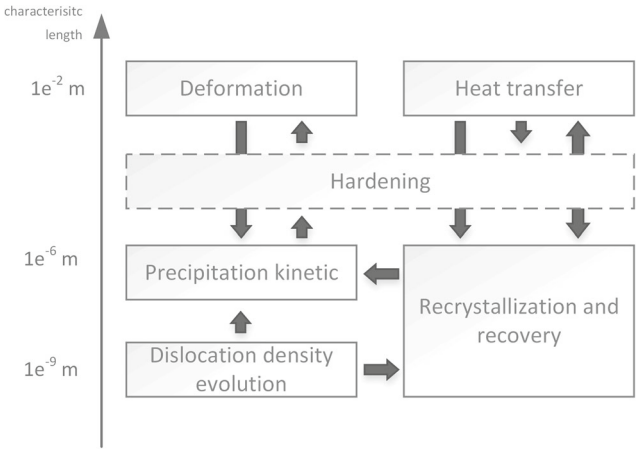

Fig. 1 The configuration of multiscale model

The case

The exemplary case introduced here after follows up on the author's previous work [4], where an AM3 structured multiscale model for TMT of metallic alloys is developed. It is composed of three main components: (i) a macroscale module for the macroscopic deformation and heat flux during thermo-mechanical treatment, typically a finite element code, (ii) a constitutive module calculating the mechanical response of the material to the evolution of user defined microscopic variables and (iii) the microsctural module(s) in charge of computing the microscopic mechanisms behind the evolution of the aforementioned internal variables.

Detailed discussion of the whole model can be found in [4]. From the perspective of this paper, it is important that the macroscopic component of the multiscale model depends on several parameters of a microstructure, including precipitations. In this work, a commercial grade aluminum alloy 6082 is annealed in isothermal condition between 250^{circ}C and 550^{circ}C. During annealing, phase transformations are expected happen. The external materials calculator MatCalc [20, 21] is integrated in the multiscale model as a microscopic that computes the precipitation kinetics of second-phase particles. It is however only one-way coupled so far: the material state and the environmental variables (temperature, heat flux) calculated by the macroscale model is passed to MatCalc, but there is no feedback. In many cases this solution is sufficient — usually, deformation happens on a much shorter time scale than precipitation kinetics. When that is not any more the case, for example during static heat treatment, a full coupling is necessary. The target structure of the multiscale model is shown in Fig. 1.

The main obstacle to a direct integration of a Thermodynamics solver for precipitation kinetics into the macroscopic model is the computation cost. In the context of finite element simulations, the precipitation model should be run for each element in the FE mesh, leading to very large computation times. It is possible to replace the Thermodynamics solver with empirical equations and the IV approach [22] to reduce the computation time, but numerous assumptions and simplifications are necessary. Alternatively, the model for precipitation may be replaced with a metamodel.

Model (step 1)

The precipitation model actually consists in an external solver — the Matcalc thermodynamic toolbox — that aims to simulate the kinetics of phase transformations based on distinct physical mechanisms, such as atomic diffusion in solids, cluster migration, vacancy formation and annihilation, interface migration, etc. Three initial material states, obtained with pre-computations were considered: as cast, homogenized and hot-rolled. The three states differ mainly by their state of precipitation, i.e. the number of phases present in their microstructure, the quantity of particles of each phase and their distribution, but also by the quantity of structural defects (dislocations, grain/subgrain boundaries, atomic vacancies) they present. Structural defects promote diffusion, therefore they play a crucial role in the kinetics of phase transformation.

In a MatCalc script, it is possible to activate/deactivate the presence of a particular phase, making it handy to simulate different initial material states.

The independent variables are the temperature and the accumulated strain, for each element of the FE mesh. The MatCalc model does not impose any restriction on the values of these conditions (in reasonable range). Contrary, development of the metamodel requires strictly defined range of input values since the training records cannot be too sparsely distributed (to keep the accuracy of the metamodel

on acceptable level) and their number is limited (due to the restricted time necessary to generate the training set). Additionally, the response of the metamodel should only feature a limited number of output variables. Hence, the model upon which it is based must be described with a precisely-defined, limited set of independent input variables — constrained by lower and upper limits — and a set of output variables. This means that the metamodel is restricted to represent a family of processes, loosing a part of the model generality. The metamodel developed in this work is able to describe a set of TMT processes, consisting of heating up, holding at high temperature and cooling down to room temperature.

Inputs and outputs(step 2)

The family of TMT processed here can be described with 7 parameters. Six parameters are necessary to describe the process itself, namely the grain size, the subgrain size, the heating and cooling rates, the isothermal holding time and the temperature history (Fig. 2a). The seventh parameter is a certain initial precipitation state that is required by the model (as cast, homogenized or hot-rolled).

Numerical simulations are often conducted only with nominal boundary conditions, the way they are described theoretically. But in a large FE computational domain, the local conditions may significantly differ from the nominal ones, mainly because of the shape of the part and the applied heating and cooling methods (e.g. the "nominal" cooling rate is 2 °C s^{-1}, while "real" cooling rates are much higher close to the sample surface and lower deep inside). Therefore, the range of parameters for the metamodel must exceed the technologically justified range. The chosen lower and upper bounds are shown in Table 1.

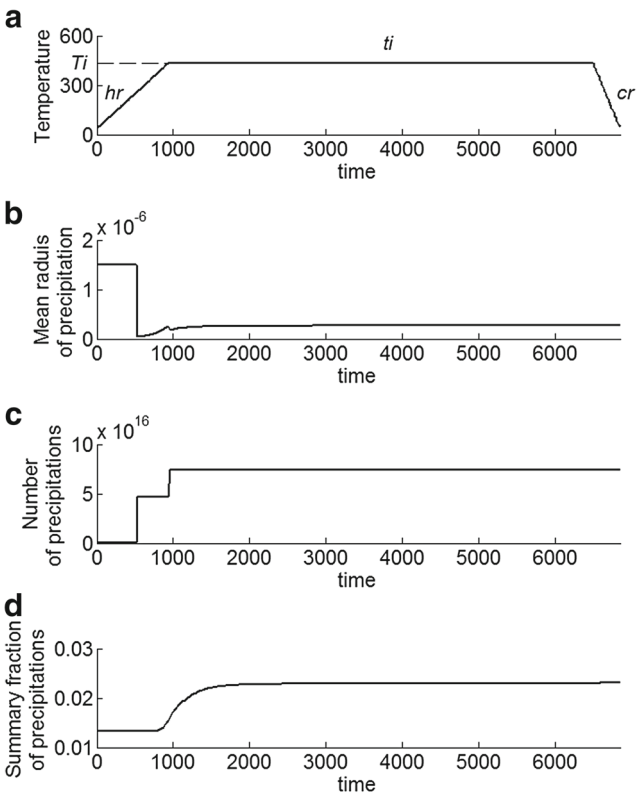

Fig. 2 The exemplary model outputs (for cast case)

Table 1 Lower and upper bounds of the input parameters

Variable	Symbol	Lower bound	Upper bound
grain size	gs	50µm	300µm
subgrain size	sgs	50µm	200µm
annealing temperature	Ti	250 °C	550 °C
heating rate	hr	0.02 °C s^{-1}	0.5 °C s^{-1}
holding time	ti	1000s	14000s
cooling rate	cr	0.01 °C s^{-1}	2 °C s^{-1}

The last issue is the output of the model. Many outputs are made available by the MatCalc simulator, of which the number of precipitates and their size are the most relevant in the context of the multiscale model described above. MatCalc outputs detailed distributions of the number of precipitates and their diameter, which cannot be used under this form in the other models (like the flow stress model), nor can be reproduced by a metamodel. In this work, the outputs from the precipitation kinetic model are: (i) the aggregated mean precipitate diameters, (ii) their phase fractions and (iii) the numbers of precipitates for all second phases, with a lower limit of 0.01µm imposed on the particle diameter. The aggregated number of precipitates N is computed with the following formula:

$$N = \sum_{p=1}^{card(P)} \sum_{i=1}^{card(Rp)} n_{p,i} \tag{4}$$

Where P is the set of all phases, R_p is the set of all precipitates of phase p, and $n_{p,i}$ counts as 1 if the i^{th} particle of the p^{th} phase has a diameter greater than 0.01µm, 0 otherwise. Exemplary output values are shown in Fig. 2b–d.

Training data (step 3)

Acquiring data for training the metamodel requires running the model numerous times. If the computational time of a single run is high, the whole procedure requires an unacceptable time. Before defining the parameter ranges to be used in a numerical experiment, the dependency of the computing time on the input parameters has been investigated. For each of the six input parameters, simulations were ran with the lower bound, the medium value and the upper bound, and that for all three initial

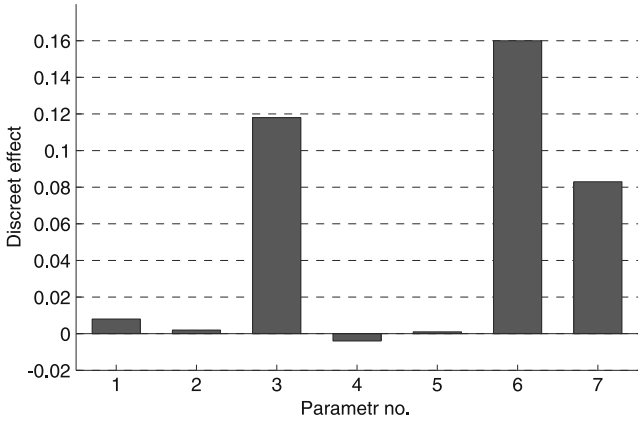

Fig. 3 Sensitivity analysis for computing time of a simulating model

material states. It was observed that the computation time was very disperse. With an Intel Core i7, most of the simulations lasted below 50s, but several runs were not completed after 3600s, which is too long for training purposes. In order to keep reasonable computation times, the range of applicability of the metamodel had to be limited. A sensitivity analysis conducted with a factorial design methodology revealed that the computation time depends mostly on the annealing temperature, the cooling rate and the initial state of the material (Fig. 3). Computations for 13 temperatures (i.e. 250, 275, 300, 325, 350, 375, 400, 425, 450, 475, 500, 525, 550 °C), 9 cooling rates (i.e. 0.01, 0.02, 0.04, 0.08, 0.15, 0.3, 0.5, 1, 2 °C s^{-1}) and the 3 material states (i.e. cast, homogenized and hot-rolled) were then computed (351 cases). On the basis of the computation time needed for those simulations, the optimal parameter range was identified with the Iterative Dichotomiser 3 (ID3) algorithm [23]. The latter is an algorithm for generating a decision tree from a dataset. The computation times were divided in two classes, *allowed* and *not allowed*, with a threshold time of 1000s. The ID3 algorithm was then used to generate a set of rules defining the boundaries of the *allowed* parameter domains. For the as cast initial state, all records were checked as *allowed*. For the initially homogenized and hot-rolled states the *allowed* domains are shown in Fig. 4.

The ID3 rules generated for the initially "homogenized" material are:

```
homogenization temperature <= 325;
OR
homogenization temperature <= 450,
cooling rate <= -0.08,
homogenization temperature > 375;
OR
homogenization temperature > 325,
cooling rate <= -0.08,
homogenization temperature <= 350;
OR
homogenization temperature > 475,
cooling rate <= -1;
OR
homogenization temperature > 325,
cooling rate > -0.04,
homogenization temperature <= 375;
OR
cooling rate <= -0.08,
homogenization temperature <= 375,
homogenization temperature > 350,
cooling rate > -0.5;
OR
cooling rate > -1,
homogenization temperature > 525,
cooling rate <= -0.15;
```

The rules for the initially hot-rolled material are:

```
homogenization temperature <= 325;
OR
homogenization temperature > 325,
cooling rate <= -0.5;
OR
```

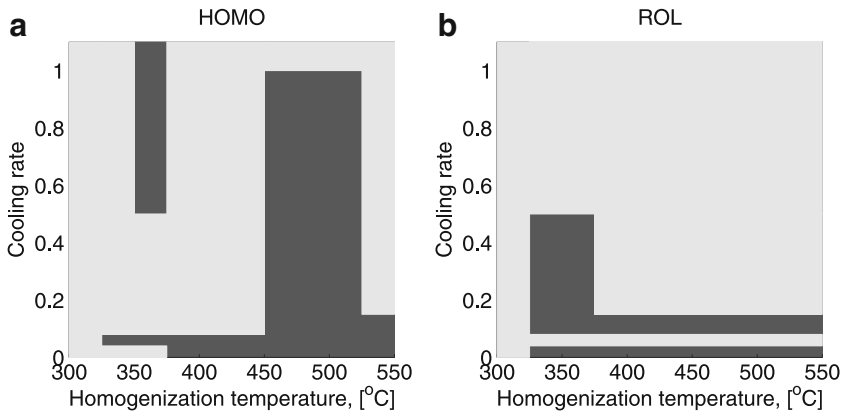

Fig. 4 The metamodel applicability ranges for "homogenized" and "hot-rolled" initial conditions.

```
cooling rate > -0.5,
cooling rate <= -0.15,
homogenization temperature > 375;
OR
homogenization temperature > 325,
cooling rate <= -0.04,
cooling rate > -0.08;
```

Due to the computational time constraints discussed above, all sets of parameters for the training dataset must fulfill the constraints imposed by these rules.

Metamodel (step 4-6)

Figure 2 shows a dynamic character of the considered model. According to the previously made assumptions the metamodel should be able to predict the values of mean radius of precipitations, number of precipitations and summary fraction of participations as a function of time. The most of metamodels described in the literature represents history-independent values (e.g. [5, 6]). In the case described in this paper, the metamodelled relationships are history-dependent, non-linear and non-differentiable. Hence, it is practically not possible to describe these values with a mathematical equation. Building a metamodel of a history-dependent, strongly varying process is a difficult task. One of the possible approaches is application of Artificial Neural Networks (ANNs). An application of ANN in microstructure evolution has been presented e.g. in [24]. The Authors used ANN to predict the stress-strain response for a unit cell with different geometry and damage parameters. Yet, the applied network had feed-forward (F-F) structure. Employing F-F ANN makes impossible to include the previous values of input signal and cannot be used as a metamodel of considered process. This problam can be solved with recurrent ANN. However, our previous works (i.e. [25]) had shown that this ANN approach is very demanding in case of recurrent network. Hence, other possibilities had been investigated.

The decision was made to use Piecewise Linear Functions (PLFs) to formulate the metamodel, as it is shown in Fig. 5. Each PLF is defined by M nodes (marked with red stars in Fig. 5). An individual PLF is established for the each

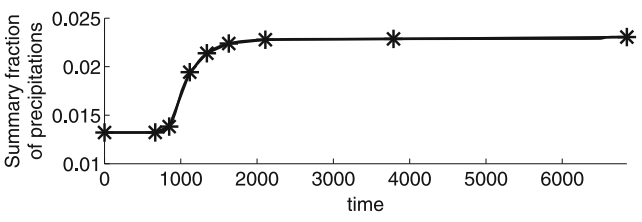

Fig. 5 The exemplary PLF with marked nodes (for $M = 9$)

output and for the each initial material state, hence finally the metamodel consisted of 9 PLFs.

A single simulation provides the record of each output as the sequences of points (t_i, v_i), where t_i is the time coordinate of the i^{th} point in the sequence, and v_i is the value coordinate of the i^{th} point in the sequence. To define the PLF the M points must be chosen to became the nodes (t_i^*, v_i^*).

Since the quality of interpolation is very sensitive to the method of points selection, a dedicated algorithm had been developed. The algorithm is composed of the preliminary step and three steps performed in the loop, see Fig. 6.

During the preliminary step the repeating points are removed and values of t_i and v_i vectors are scaled into the range $[-1, 1]$. Next the initial set of nodes is established. The first and last points in the profile became the end-nodes. The mid-nodes are evenly distributed between the end-nodes. In each iteration the mid-nodes are moved one by one. Three forms of move can be applied. The base one is replacing of the point already selected to be the node by its left and right neighbors. After replacing of the single node, the approximation error is calculated as the integral of the absolute value of the difference between the values of PLF and the values of the model output using trapezoidal method. If the error is reduced, the new point became the node.

If none of the nodes are moved then the second way applies, moving nodes which are already lined on a straight line. For an each mid-node, coefficients of second order polynomial are computed based on the node and its two neighbors. If the coefficient of the second power is less then the assumed threshold, the node is moved to the middle of the PLF section characterized by the greatest approximation error. The similar displacement is made for nodes lying too close to each other (the third form of node movement). The algorithm stops when none of the nodes has been moved during the last iteration or after exceeding the maximal number of iteration. The approximation had been performed for several numbers of nodes, varying from 6 to 10. On the ground of this preliminary results, the number of nodes was established to $M = 9$.

When the nodes are selected, the relation between their coordinates (t_i^*, v_i^*) and the input parameters must be defined. There are several approximation methods available. The feedforward ANNs had been firstly considered, however due to the relatively small number of available training records (2071 in "casted" set, 825 in "homogenized" set and 779 in "hot-rolled" set) the Kriging method was chosen [13]. Kriging is based on the idea that a value in a given point can be estimated on the basis of an average of known values in the neighboring points, assuming that the influences of these points are proportional to the distance to the considered point. In other words, the approximation procedure has to follow the trends of the experimental

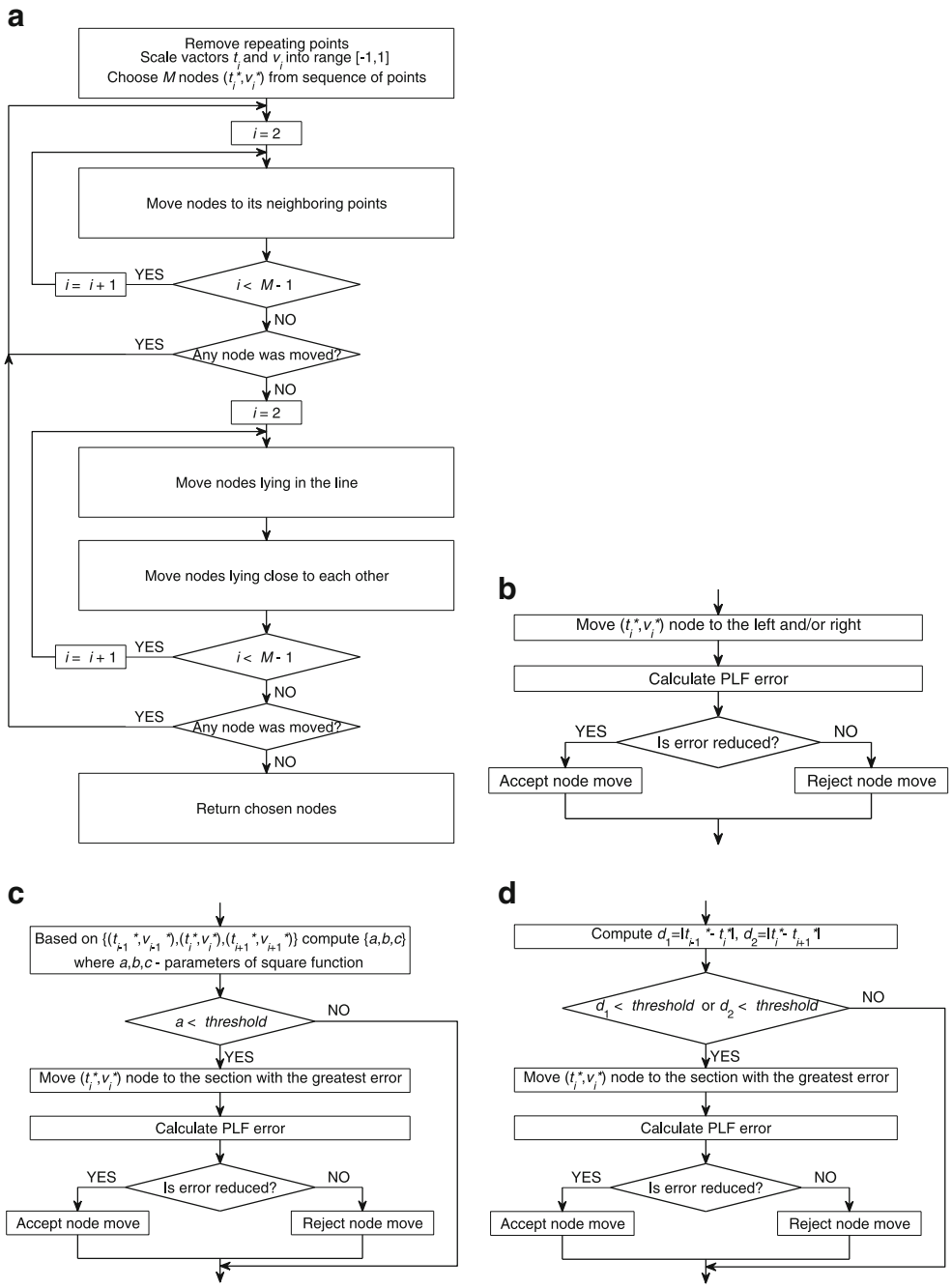

Fig. 6 Flowchart of the nodes selection algorithm. **a** main diagram; (**b**, **d**) detailed diagrams of: **b** moving nodes to its neighboring points, **c** moving nodes lying in the line and **d** moving nodes lying close to each other

data and the surrogate function should increase when such increase is expected for an increment of the variables [26]. The definition of the Kriging metamodel requires selection the regression and correlation models. The most commonly used regression models are low order polynomials. As correlation model the exponential, gauss, linear, spherical, cubic or spline function can be applied. Within this paper, the zero order polynomial and gauss function were used as the regression and correlation model, respectively.

As mentioned above, the metamodel consists 9 PLFs, each PLF is defined by 9 nodes and the each node has two coordinates. That requires 162 Kriging approximators. However, the coordinates of the first node in each PLF are known a'priori (the time equal to 0 and the value equal

to the corresponding output from simulation model), the final number of necessary Kriging approximations is equal to 142. To improve the approximation accuracy, only the inputs with the significant influence are taken into account. The influence was examined using SA. The results are shown in Fig. 7a. For the each output variable (mean radius of precipitation, number of precipitation and summary fraction of precipitations), there are 9 time coordinates and 9 values. The dark square means that the specific node coordinate (time or value) is not influenced by the specific input, e.g. analyzing the second row from the top it can be seen that for mean radius of precipitation time coordinate of the second node in "casted" case depends only on the heating rate (hr), while in the "homogenized" case it depends on all inputs except for the cooling rate (cr).

Due to the occurance of zero values in the testing set the Kriging approximations were evaluated by the absolutive error described by following formula:

$$\varepsilon^* = \frac{1}{N}\sqrt{\sum_{i=1}^{N}\left(\widetilde{x_i^*} - x_i^*\right)^2} \tag{5}$$

where: N is the number of testing records, $\widetilde{x_i^*}$ is the time ($\widetilde{t_i^*}$) or value ($\widetilde{v_i^*}$) coordinate of the i^{th} node computed by the

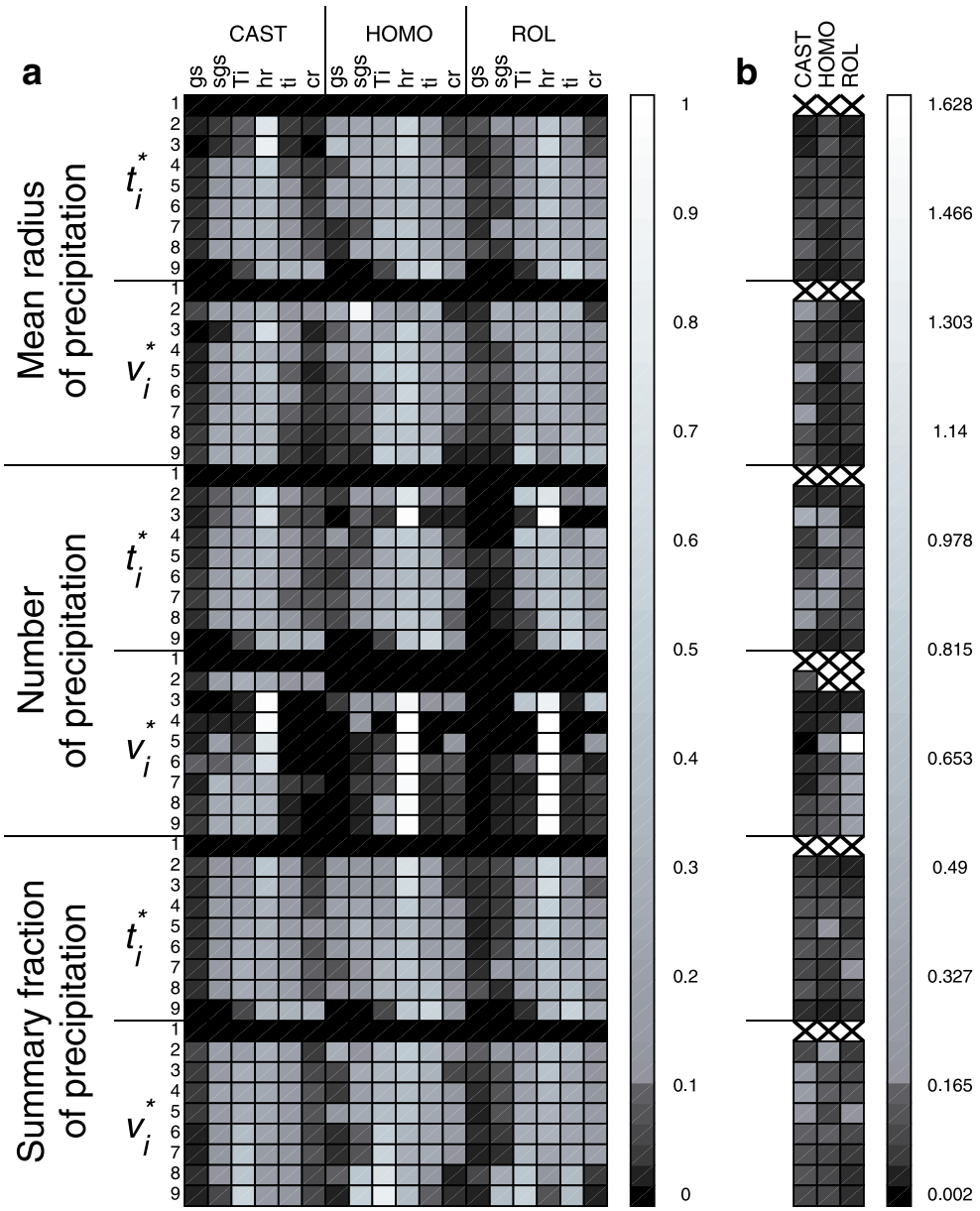

Fig. 7 The SA results (**a**) and the errors of Kriging approximators (**b**)

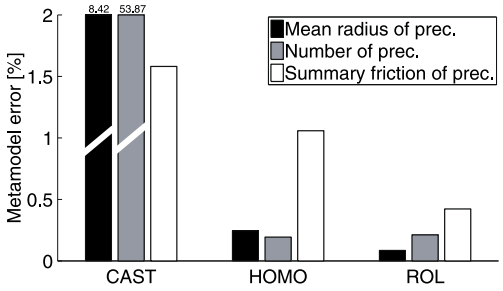

Fig. 8 Metamodel error

Kriging approximator, x_i^* is the exact respective value. The obtained errors are presented in Fig. 7b. The main impact on the obtained errors has the number of training points. Due to long computing times, the number of available records was equal to 2071, 825 and 779 for casted, homogenized and hot-rolled material state, respectively. The performed Sensitivity Analysis reviles the model inputs which have negligible influence on the output value and thus, they weren't took into account during building the metamodel. It makes metamodel less complicated but also it reduces the number of training records. In some cases the number of records used for training was reduced to 239. The mean error of all Kriging approximators is equal to 0.129. Taking into account that the values of x_i^* lie within the range $[-1, 1]$ the error seems to be high. However, as it was presented in [13], the error can be reduced by increasing the number of records in training set.

Results (step 7)

The metamodel accuracy was evaluated using testing datasets consisted of model simulations which was not used during the training. Errors were calculated separately for the each PLF using the equation:

$$\varepsilon^{PLF} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{n^i}\sum_{j=1}^{n^i}\frac{\left|PLF\left(t_j^i\right) - v_j^i\right|}{\overline{v^i}} \tag{6}$$

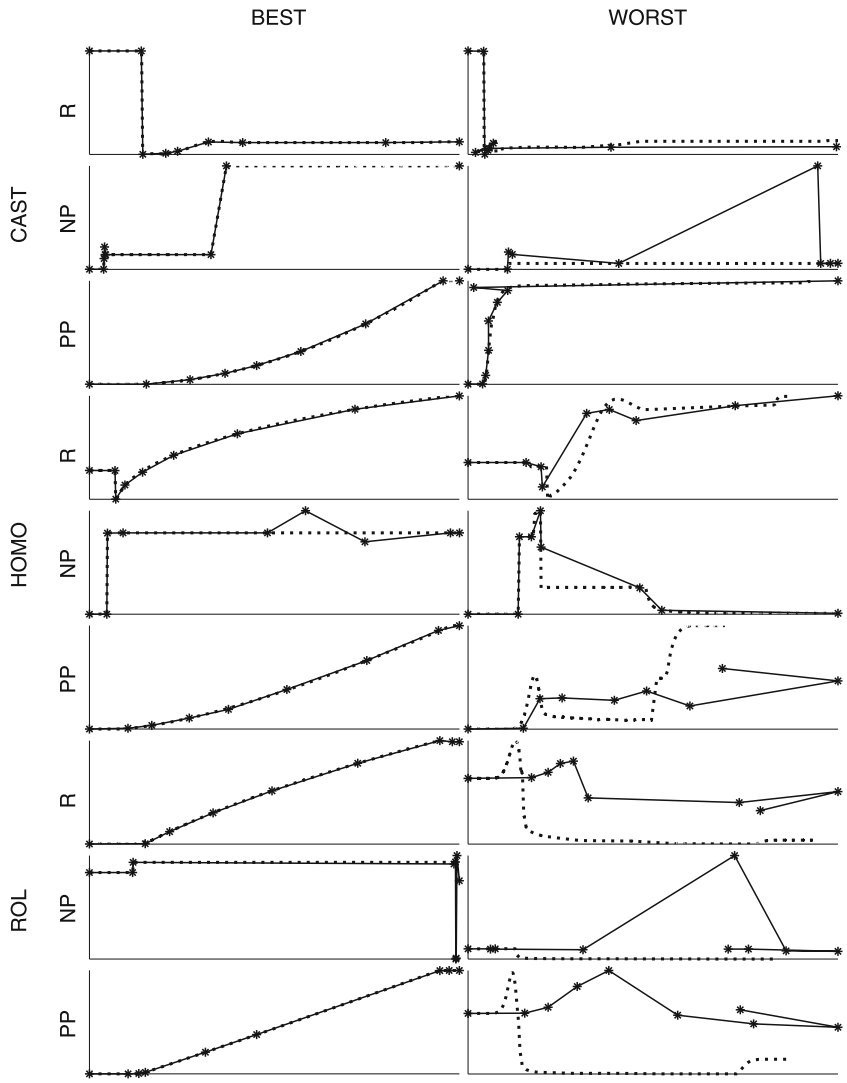

Fig. 9 Exemplary metamodel outputs

where: N is the number of testing records, n^i is the number of the points in the i^{th} sequence, (t^i_j, v^i_j) is j^{th} point in the i^{th} sequence and $\overline{v^i}$ is the mean value in the i^{th} sequence. The values of each PLF error are presented in Fig. 8. The best and the worst cases from the testing set are presented in Fig. 9

Metamodelling is aimed at replacing of a model with a metamodel, which is *much faster* and *sufficiently accurate*. The first criterion is fulfilled indisputably – the computing time of the metamodel is few orders of magnitude smaller than the computing time of model. Accuracy of the metamodel requires more thorough discussion. If mean errors of PLFs are analyzed, the error level is close to 2%, which is very good results. However, analysis of best/worse cases shows less satisfying results. As expected, the most of the bests cases shows almost perfect fitting of approximated values to simulated ones. However, some distribuances are present (5th curve in Fig. 9). If the value itself is analyzed (number of precipitates in this case), result is fully satisfactory, however if the metamodel would be used to calculate a goal function in optimization procedure, such inconsistency might be a source of some problems.

Analysis of the worst cases shows, that even relatively small error value does not guarantee reliable representation of model. It can be noticed, that for several cases the result of metamodel are completely wrong, missing both values and trends (e.g. 8th curve in Fig. 9). The second important issue is an erroneous character of some curves with time coordinate of last point lower then at least one of other points. That shows some deficiency of developed algorithm of PLF representation of model outputs, in which each output variable is evaluated separately. The main source of inaccuracy of the created metamodels is the small number of training data. The error can be reduced by performing more model simulations and re-training the Kriging approximators.

Discussion and conclusions

The obtained results shows that developed methodology of metamodelling can lead to successful replacement of a model with a metamodel. Low values of the mean metamodelling error indicates very good reliability of PLF-based approximation. However, the present form of the algorithm does not eliminate the situation, when for some specific combinations of input parameters, results are completely wrong. This is a significant drawback and the proper arrangements will be included to the algorithm in future. One of the most important issues is detection of PLF approximation not being the functions (e.g. 6th to 9th curves in Fig. 9). The advantage arising from embedding of metamodels in AM3 framework is that the metamodel does not have to be valid in whole domain. Instead, knowledge-based controlling module of multiscale model can be enriched with rules denying application of the metamodel in some sub-domains, replacing it back with model itself. Besides, it is expected that increasing the number of model computations will significantly increase both the reliability and possible range of application of metamodels. That will be achieved with application of High Performance Computing techniques, planned for the near future.

Acknowledgements This work is supported by the NCN, project No. 2011/01/D/ST8/04984 MatCalc model of precipitation kinetics was provided by Cecilia Poletti and Romain Bureau, Institute of Materials Science, Joining and Forming, Graz University of Technology. Financial support by the Austrian Federal Government (in particular from Bundesministerium für Verkehr, Innovation und Technologie and Bundesministerium für Wissenschaft, Forschung und Wirtschaft) represented by Sterreichische Forschungsförderungsgesellschaft mbH and the Styrian and the Tyrolean Provincial Government, represented by Steirische Wirtschaftsförderungsgesellschaft mbH and Standortagentur Tirol, within the framework of the COMET Funding Program is gratefully acknowledged.

Compliance with Ethical Standards

Conflict of interests This study was funded by Narodowe Centrum Nauki (project No. 2011/01/D/ST8/04984). The authors declare that they have no conflict of interest.

References

1. Maciol P, Gotfryd L, Macioł A (2012) Knowledge based system for runtime controlling of multiscale model of ion-exchange solvent extraction. In: AIP Conference Proceedings, ICNAAM 2012: International Conference of Numerical Analysis and Applied Mathematics, Vol. 1479, American Institute of Physics, Kos, Greece, pp 125–128. https://doi.org/10.1063/1.4756078. http://link.aip.org/link/?APCPCS/1479/125/1
2. Macioł P, Krumphals A, Jędrusik S, Macioł A, Sommitsch C (2013) Rule-based expert system application to optimizing of multiscale model of hot forging and heat treatment of Ti-6Al-4V. In: Idlesohn S., Papadrakakis E., Schrefler B. (eds) V International Conference on Computational Methods for Coupled Problems in Science and Engineering COUPLED PROBLEMS 2013, Ibiza, pp 1237–1248
3. Macioł P, Bureau R, Sommitsch C (2014) An object-oriented analysis of complex numerical models. Key Eng Mater 611-612:1356–1363. http://www.scientific.net/KEM.611-612.1356
4. Macioł P, Bureau R, Poletti C, Sommitsch C, Warczok P, Kozeschnik E (2015) Agile multiscale modelling of the thermo-mechanical processing of an aluminium alloy. In: Key Engineering Materials, Vol. 651-653, pp 1319–1324. http://www.scientific.net/KEM.651-653.1319

5. Rauch L, Kuziak R, Pietrzyk M (2014) From high accuracy to high efficiency in simulations of processing of dual-phase steels. Metallurgical and Materials Transactions B 45(2):497–506. https://doi.org/10.1007/s11663-013-9926-5
6. Rauch L, Szeliga D, Bachniak D, Bzowski K, Słota R, Pietrzyk M, Kitowski J (2015) Identification of multi-inclusion statistically similar representative volume element for advanced high strength steels by using data farming approach. Procedia Comput Scie 51:924–933. https://doi.org/10.1016/j.procs.2015.05.227. http://www.sciencedirect.com/science/article/pii/S1877050915010352
7. Pietrzyk M (1994) Numerical aspects of the simulation of hot metal forming using internal variable method. Metallurgy Foundry Eng 20:423–439
8. Takaki T, Yamanaka A, Higa Y, Tomita Y (2007) Phase-field model during static recrystallization based on crystal-plasticity theory. J Computer-Aided Mater Des 14(S1):75–84. https://doi.org/10.1007/s10820-007-9083-8
9. Sen O, Davis S, Jacobs G, Udaykumar H (2015) Evaluation of convergence behavior of metamodeling techniques for bridging scales in multi-scale multimaterial simulation. J Comput Phys 294:585–604. https://doi.org/10.1016/j.jcp.2015.03.043. http://linkinghub.elsevier.com/retrieve/pii/S0021999115001989
10. Saltelli A, Chan K, Scott EM (2009) Sensitivity Analysis. https://books.google.pl/books/about/Sensitivity_Analysis.html?id=gOcePwAACAAJ&pgis=1
11. Szeliga D. (2013) Identification problems in metal forming. A comprehensive study, AGH University of Science and Technology Press
12. Myers RH, Montgomery DC, Anderson-Cook CM (2009) Response surface methodology: process and product optimization using designed experiments. https://books.google.pl/books/about/Response_Surface_Methodology.html?id=89oznEFHF_MC&pgis=1
13. Kusiak J, Sztangret Ł, Pietrzyk M (2015) Effective strategies of metamodelling of industrial metallurgical processes. Adv Eng Softw 89:90–97. https://doi.org/10.1016/j.advengsoft.2015.02.002
14. Hambli R (2011) Multiscale prediction of crack density and crack length accumulation in trabecular bone based on neural networks and finite element simulation. International Journal for Numerical Methods in Biomedical Engineering 27(4):461–475. https://doi.org/10.1002/cnm.1413
15. Hambli R, Barkaoui A Multiscale approach for bone remodeling simulation based on finite element and neural network computation. arXiv:1107.3817
16. Górecki G, Rauch Ł, Pietrzyk M (2014) Ann-based metamodeling with output values clusterization as an approach to robust inverse analysis. Comput Methods Mater Sci 14:167–179
17. Pietrzyk M, Kusiak J, Szeliga D, Rauch Ł, Sztangret Ł, Górecki G (2016) Application of metamodels to identification of metallic materials models. Adv Mater Sci Eng 2016:1–20. https://doi.org/10.1155/2016/2357534. http://www.hindawi.com/journals/amse/2016/2357534/
18. Morris MD (1991) Factorial Sampling Plans for Preliminary Computational Experiments. Technometrics 33(2):161–174. https://doi.org/10.2307/1269043. http://www.jstor.org/stable/1269043
19. Szeliga D, Macioł P, Sztangret Ł (2017) Substituting of a thermodynamic simulation with a metamodel in the scope of multiscale modelling. In: THERMEC 2016, Vol. 879 of Materials Science Forum, Trans Tech Publications, pp 1207–1212. https://doi.org/10.4028/www.scientific.net/MSF.879.1207
20. Kozeschnik E, Bataille C, Janssens K (2013) Modeling Solid-State Precipitation. https://books.google.pl/books/about/Modeling_Solid_State_Precipitation.html?id=bm4ERTzdQxAC&pgis=1
21. Kozeschnik E, Svoboda J, Fratzl P, Fischer F, Fratzl P, Kozeschnik E (2004) Modelling of kinetics in multi-component multi-phase systems with spherical precipitates. Mater Sci Eng A 385(1-2):166–174. https://doi.org/10.1016/j.msea.2004.06.016. http://www.sciencedirect.com/science/article/pii/S0921509304008202. http://www.sciencedirect.com/science/article/pii/S0921509304008214
22. Dutta B, Sellars CM (1987) Effect of composition and process variables on Nb(C, N) precipitation in niobium microalloyed austenite. J Mater Sci Technol 3(3):197–206. https://doi.org/10.1179/026708387790122846
23. Quinlan JR (1986) Induction of decision trees. Mach Learn 1(1):81–106. https://doi.org/10.1023/A:1022643204877. http://link.springer.com/10.1007/BF00116251
24. Rami H-J, Pecknold DA, Ghaboussi J, Voyiadjis GZ (2001) Simulated micromechanical models using artificial neural networks. J Eng Mech 127(7):730–738
25. Kusiak J, Sztangret L, Rauch L, Pietrzyk M (2014) Metamodel driven optimization of thermomechanical industrial processes. Comput Methods Mater Sci 14(1):20–26
26. Sacks J, Welch WJ, Mitchell TJ, Wynn HP (1989) Design and Analysis of Computer Experiments. Stat Sci 4(4):409–423. https://doi.org/10.1214/ss/1177012413